Yokohama Publishers
ISSN 1349-8169 ONLINE JOURNAL

Pacific Journal of Optimization
Vol. 21, No. 1, pp. 125–136, 2025

A DECOMPOSITION-BASED DOMAIN-CUT ALGORITHM FOR MIXED INTEGER PROGRAMMING PROBLEMS WITH SCENARIO CONSTRAINTS*

Fenlan Wang

Abstract: A new approach is presented for exactly solving linear mixed integer programming problems with scenario constraints in this paper. We first decompose the original problem into two subproblems, then cut off some integer boxes which do not contain any optimal solution via domain partition techniques. Integrating the solution scheme into a branch-and-bound frame, the proposed solution method reduces the optimality gap gradually in the solution iterations. Furthermore, the proposed solution method is of a finite-step convergence. Finally, the numerical experiments are also given and the comparison results with the traditional branch-and-bound method show the proposed method is very efficient.

Key words: *mixed integer programming, optimality gap, branch-and-bound, scenario constraints, domain partition*

Mathematics Subject Classification: *90C11, 90C05*

1 Introduction

We consider in this paper the following class of linear mixed integer programming problems:

$$
\begin{array}{lll}
(P) & \min & f(x) = c^T x \\
 & \text{s.t.} & T^s x + W^s y^s \geq h^s,\ s = 1, 2 \cdots, S \\
 & & y^s \geq 0, \\
 & & Dx \leq d, \\
 & & X = \{x |\ l \leq x \leq u,\ \text{integer}\}
\end{array}
$$

where c is an n-dimensional column vector, $T^s \in \mathbb{R}^{m_1 \times n}$, $h^s \in \mathbb{R}^{m_1}$, $y^s \in \mathbb{R}^{n_1}$ and $W^s \in \mathbb{R}^{m_1 \times n_1}$ for each scenario $s \in \{1, 2 \cdots, S\}$. $D = (d_{ij})_{m \times n}$ is an $m \times n$ matrix, d is an m-dimensional column vector, $l = (l_1, \ldots, l_n)^T \in \mathbb{Z}^n$ and $u = (u_1, \ldots, u_n)^T \in \mathbb{Z}^n$ are the lower and upper bounds of x, respectively.

Integer programming models have many applications in engineering such as capital budgeting [23], capacity planning [7], portfolio selection problem with integer lots [32]. Now there have been many methods in literature for solving integer programming problems

*This research was partially supported by the National Natural Science Foundation of China under Grants 12071215 and by Key Laboratory of Mathematical Modelling and High Performance Computing of Air Vehicles (NUAA), MIIT, Nanjing 211106, China.

 DOI: https://doi.org/10.61208/pjo-2023-031

[19, 1, 2, 10, 11, 26, 27], most of which are dynamic programming-based methods and continuous relaxation-based branch-and-bound methods or a combination of dynamic programming method and branch-and-bound method. The dynamic programming method is used to solve separable integer programming problems with a single constraint [3, 12, 16]. Due to the property 'curse of dimensionality' of the dynamic programming, it is not easy to be extended directly to solve multiple constrained separable integer programming problems. Branch-and-bound methods based on the continuous relaxation problem are often used for solving convex integer programming problems, since its continuous relaxation problems can be solved easily [5, 8, 13, 17, 18, 23, 28]. Also branch-and-bound methods based on global optimization over a polyhedron for concave integer programming problems were presented in [4, 6, 9, 14, 15, 6]. Hybrid approach which combines the dynamic programming method with the branch-and-bound method was presented in [22, 24, 25] for solving nonlinear separable integer programming problems.

In this paper, we consider linear mixed integer programming problem (P) with scenario constraints, which is a special case of linear mixed integer programming problems and can cover many problems in practice such as stochastic or deterministic two stage linear programming problems. As we know, problem (P) can be solved by the traditional branch-and-bound method. However, in problem (P), the total number of variables is $n + n_1 \times S$ where there are n integer variables and $n_1 \times S$ continuous variables, and the number of the constraints is $m + m_1 \times S$ except the constraint X and $y^s \geq 0$. Therefore, problem (P) is a kind of large-scale linear mixed integer programming problems. Also the traditional branch-and-bound method is not very efficient for problem (P) since too much storage space is needed and it will spend too much CPU time to solve the continuous relaxation problem in the traditional branch-and-bound method. This paper aims at developing a new exact algorithm to efficiently solve problem (P).

Recently, some domain cut skills are exploited to solve some integer programming problems. An novel contour cut technique was presented in [20] for separable quadratic integer programming problems. Another new domain cut technique is proposed in [29] for solving separable integer programming problems with concave objective function and linear constraints. A new algorithm is presented in [31] for general separable integer programming problems, which combines linear approximation and Lagrangian dual with a simple cut. A successive domain-reduction scheme for linearly constrained quadratic integer programming problems is also studied in [30]. Motivated by these novel domain cut techniques, we develop a new algorithm combining decomposition method with domain reduction technique for (P) in the paper. First we decompose the large-scale problem (P) into small ones according to the characteristics of (P) and the domain reduction technique shrinks the feasible region gradually in the iteration process. Thus the optimal solution of (P) can be found quickly within a finite steps of iterations. Finally, we give the numerical experiments and the comparison results with the traditional branch-and-bound method.

The paper is organized as follows. Section 2 gives the decomposition method of (P) and the feasibilities of the scenario constraints are discussed. Domain cut schemes are presented in Section 3. Section 4 describes the proposed algorithms in details. Preliminary computational results are reported in Section 5 for randomly generated linear mixed integer programming problems. Finally, conclusion is given in Section 6.

2 Decomposition of the subproblem

For any α and $\beta \in \mathbb{Z}^n$, we first introduce two definitions: Let $\alpha = (\alpha_1, \alpha_2, \cdots, \alpha_n)$ and $\beta = (\beta_1, \beta_2, \cdots, \beta_n)$, if $\alpha_i \leq \beta_i$, for all $i = 1, 2, \cdots, n$, then $\alpha \leq \beta$, Otherwise, $\alpha \not\leq \beta$. Let $[\alpha, \beta]$

be the box (hyper-rectangle) formed by α and β, $[\alpha, \beta] = \{x \mid \alpha_j \leq x_j \leq \beta_j, j = 1, \ldots, n\}$. Denote by $\langle \alpha, \beta \rangle$ the set of integer points in $[\alpha, \beta]$,

$$\langle \alpha, \beta \rangle = \{x \mid \alpha_j \leq x_j \leq \beta_j,\ x_j \text{ integer},\ j = 1, \ldots, n\} = \Pi_{j=1}^{n} \langle \alpha_j, \beta_j \rangle.$$

Set $\langle \alpha, \beta \rangle$ is termed an integer box. For convenience, we define $[\alpha, \beta] = \langle \alpha, \beta \rangle = \emptyset$ if $\alpha \not\leq \beta$. Let $\langle \alpha, \beta \rangle \subseteq X = \{x \mid l_j \leq x_j \leq u_j,\ x_j \text{ integer},\ j = 1, \ldots, n\}$ be a nonempty integer box and $(P_{\langle \alpha, \beta \rangle})$ be a subproblem of (P) by replacing X by $\langle \alpha, \beta \rangle$ with $\alpha \leq \beta$. Then we consider the following subproblem:

$$\begin{array}{llll} (P_{\langle \alpha, \beta \rangle}) & \min & f(x) = c^T x \\ & \text{s.t.} & T^s x + W^s y^s \geq h^s,\ \forall s \in \{1, 2, \cdots, S\} \\ & & Dx \leq d \\ & & x \in \langle \alpha, \beta \rangle \\ & & y^s \geq 0,\ \forall s \in \{1, 2, \cdots, S\} \end{array}$$

We can decompose the subproblem $(P_{\langle \alpha, \beta \rangle})$ into two parts: main problem (MP) and the feasibility subproblems (SP_s) of $s \in \{1, 2, \cdots, S\}$ scenario constraints.

$$\begin{array}{lll} (MP) & \min & f(x) = c^T x \\ & \text{s.t.} & Dx \leq d, \\ & & x \in \langle \alpha, \beta \rangle \end{array}$$

Obviously, the optimal objective function value of the continuous relaxation problem of problem (MP) can provide a lower bound for problem $(P_{\langle \alpha, \beta \rangle})$ by discarding the integer limitation.

For a given feasible solution $\hat{x}$ of (MP), we need to determine whether $\hat{x}$ is feasible to the scenario constraints or not. For $\forall s \in \{1, 2, \cdots, S\}$, we can do this by solving the following subproblems:

$$\begin{array}{lll} (SP_s) & \min & t^s \\ & \text{s.t.} & T^s \hat{x} + W^s y^s + e t^s \geq h^s, \\ & & y^s \geq 0,\ t^s \geq 0 \end{array}$$

where $e \in \mathbb{R}^{m_1}$ is a vector of 1's. Obviously, if $(t^s)^* = 0$ where $(t^s)^*$ is the optimal solution of problem (SP_s), then $\hat{x}$ is feasible to the constraint $T^s x + W^s y^s \geq h^s$. Otherwise, it is infeasible. And we have the dual problem of (SP_s):

$$\begin{array}{lll} (DP_s) & \max & (h^s - T^s \hat{x})^T \mu^s \\ & \text{s.t.} & (W^s)^T \mu^s \leq 0, \\ & & e^T \mu^s \leq 1 \\ & & \mu^s \geq 0 \end{array}$$

Due to the strong duality property of linear programming problem, if $(h^s - T^s \hat{x})^T (\mu^s)^* = 0$ where $(\mu^s)^*$ is the optimal solution of (DP_s), then $\hat{x}$ is feasible to the constraint $T^s x + W^s y^s \geq h^s$; If $(h^s - T^s \hat{x})^T (\mu^s)^* > 0$, it is infeasible.

$\boxed{3}$ Domain partition

In this section, we will present the domain partition techniques for several cases according to whether an integer point $\hat{x}$ being feasible to each constraint or not. The following lemmas show us how to cut off a subbox from an integer box $\langle \alpha, \beta \rangle$ associated with the subproblem.

Case 1. $\hat{x}$ is feasible to $(P_{\langle \alpha, \beta \rangle})$.

Lemma 3.1. *If $\hat{x}$ is feasible to $(P_{\langle \alpha, \beta \rangle})$, then there is no solution better than $\hat{x}$ in the integer box $B_1 = \langle \bar{\alpha}, \bar{\beta} \rangle$ which is determined by the following equations:*

$$\begin{aligned} \bar{\alpha}_j &= \begin{cases} \hat{x}_j, & c_j > 0 \\ \alpha_j, & c_j < 0 \end{cases}, \quad j = 1, 2, \cdots, n; \\ \bar{\beta}_j &= \begin{cases} \beta_j, & c_j > 0 \\ \hat{x}_j, & c_j < 0 \end{cases}, \quad j = 1, 2, \cdots, n. \end{aligned} \tag{3.1}$$

Proof. For $\forall x \in \langle \bar{\alpha}, \bar{\beta} \rangle$, $\bar{\alpha}_j \leq x_j \leq \bar{\beta}_j$, $j = 1, 2, \ldots, n$. From Equation (3.1), we have $c_j(x_j - \hat{x}_j) \geq 0$, for $j = 1, 2, \ldots, n$. Thus, $c^T(x - \hat{x}) \geq 0$, $c^T x \geq c^T \hat{x}$. That is, there is no solution better than $\hat{x}$ in the integer box B_1. □

Case 2. $\hat{x}$ is an integer solution but not feasible to some $s \in \{1, 2, \cdots, S\}$ scenario constraint.

Lemma 3.2. *If $\hat{x}$ is infeasible to some $s \in \{1, 2, \cdots, S\}$ scenario constraint, then there is no feasible solution in the integer box $B_2 = \langle \bar{\alpha}, \bar{\beta} \rangle$ which is determined by the following equation:*

$$\begin{aligned} \bar{\alpha}_j &= \begin{cases} \hat{x}_j, & ((T^s)^T \lambda^s)_j < 0 \\ \alpha_j, & ((T^s)^T \lambda^s)_j > 0 \end{cases}, \quad j = 1, 2, \cdots, n; \\ \bar{\beta}_j &= \begin{cases} \beta_j, & ((T^s)^T \lambda^s)_j < 0 \\ \hat{x}_j, & ((T^s)^T \lambda^s)_j > 0 \end{cases}, \quad j = 1, 2, \cdots, n. \end{aligned} \tag{3.2}$$

where λ^s is the optimal solution of problem (DP_s).

Proof. For $\forall x \in \langle \bar{\alpha}, \bar{\beta} \rangle$, $\bar{\alpha}_j \leq x_j \leq \bar{\beta}_j$, $j = 1, 2, \ldots, n$. From Equation (3.2), we have $((T^s)^T \lambda^s)_j (x_j - \hat{x}_j) \leq 0$, for $j = 1, 2, \ldots, n$. Thus, $((T^s)^T \lambda^s)^T (x - \hat{x}) \leq 0$, $(h^s - T^s x)^T \lambda^s \geq (h^s - T^s \hat{x})^T \lambda^s > 0$, which shows that there is no feasible solution in the integer box B_2. □

Case 3. $\hat{x}$ is not feasible to $Dx \leq d$.

Lemma 3.3. *If $D\hat{x} \not\leq d$, without loss of generality, assume $D_i \hat{x} > d_i$, where $D_i = (d_{i1}, d_{i2}, \ldots, d_{in})$. Then there is no feasible solution in the integer box $B_3 = \langle \bar{\alpha}, \bar{\beta} \rangle$ which is determined by the following equations:*

$$\begin{aligned} \bar{\alpha}_j &= \begin{cases} \hat{x}_j, & (D_i)_j > 0 \\ \alpha_j, & (D_i)_j < 0 \end{cases}, \quad j = 1, 2, \cdots, n; \\ \bar{\beta}_j &= \begin{cases} \beta_j, & (D_i)_j > 0 \\ \hat{x}_j, & (D_i)_j < 0 \end{cases}, \quad j = 1, 2, \cdots, n. \end{aligned} \tag{3.3}$$

Proof. For $\forall x \in \langle \bar{\alpha}, \bar{\beta} \rangle$, $\bar{\alpha}_j \leq x_j \leq \bar{\beta}_j$, $j = 1, 2, \ldots, n$. From (3.3), we have $(D_i)_j (x_j - \hat{x}_j) \geq 0$, for $j = 1, 2, \ldots, n$. Thus, $D_i(x - \hat{x}) \geq 0$, $D_i x \geq D_i \hat{x} > d_i$. So for $\forall x \in B_3$, $D_i x \geq d_i$. □

Case 4. $\hat{x}$ is not an integer solution, but the continuous optimal solution to (MP).

First we can obtain an integer point $\hat{x}^-$ as follows:

$$(\hat{x}^-)_j = \begin{cases} \lfloor \hat{x}_j \rfloor, & c_j > 0 \\ \lceil \hat{x}_j \rceil, & c_j < 0 \end{cases}, \quad j = 1, 2, \cdots, n$$

where $\lceil \hat{x}_j \rceil$ and $\lfloor \hat{x}_j \rfloor$ represent the ceiling and the floor of $\hat{x}_j$, respectively.

Then we will have the following lemma to find the corresponding domain cut.

Lemma 3.4. *There is no feasible solution in the integer box $B_4 = \langle \bar{\alpha}, \bar{\beta} \rangle$ which is determined by the following equation:*

$$\begin{aligned} \bar{\alpha}_j &= \begin{cases} (\hat{x}^-)_j, & c_j < 0 \\ \alpha_j, & c_j > 0 \end{cases}, \quad j = 1, 2, \cdots, n; \\ \bar{\beta}_j &= \begin{cases} \beta_j, & c_j < 0 \\ (\hat{x}^-)_j, & c_j > 0 \end{cases}, \quad j = 1, 2, \cdots, n. \end{aligned} \tag{3.4}$$

Proof. Suppose $\exists\ x^0 \in B_4$ is feasible to $P_{\langle \alpha, \beta \rangle}$. Since $x^0 \in \langle \bar{\alpha}, \bar{\beta} \rangle$, $\bar{\alpha}_j \le x^0_j \le \bar{\beta}_j$, $j = 1, 2, \ldots, n$. From (3.4), we have $c_j(x^0_j - (\hat{x}^-)_j) \le 0$ and $c_j((\hat{x}^-)_j - \hat{x}_j) \le 0$, for $j = 1, 2, \ldots, n$. Thus, $c^T(x^0 - \hat{x}^-) \le 0$ and $c^T(\hat{x}^- - \hat{x}) \le 0$. $c^T x^0 \le c^T \hat{x}^- \le c^T \hat{x}$ which is a contradiction with $\hat{x}$ being the continuous optimal solution of (MP). □

Then $B_1,\ B_2,\ B_3,\ B_4$ can be cut off from $\langle \alpha, \beta \rangle$ without removing off any feasible solution better than $\hat{x}$.

After we cut off a subbox $\langle \gamma, \delta \rangle$ from $\langle \alpha, \beta \rangle$, the remaining area is not a hyper-rectangle in general. For the sake of further cutting the domain, the revised region needs to be partitioned into a union of integer subboxes. The following lemma tells us how to partition the domain.

Lemma 3.5 ([20, 21])**.** *Let $A = \langle \alpha, \beta \rangle$ and $B = \langle \gamma, \delta \rangle$, where $\alpha,\ \beta,\ \gamma,\ \delta \in \mathbb{Z}^n$ and $\alpha \le \gamma \le \delta \le \beta$. Then*

$$\begin{aligned} A \setminus B \ &= \ \{\cup_{j=1}^n \left(\Pi_{i=1}^{j-1} \langle \alpha_i, \delta_i \rangle \times \langle \delta_j + 1, \beta_j \rangle \times \Pi_{i=j+1}^n \langle \alpha_i, \beta_i \rangle \right) \} \\ & \quad \cup \{\cup_{j=1}^n \left(\Pi_{i=1}^{j-1} \langle \gamma_i, \delta_i \rangle \times \langle \alpha_j, \gamma_j - 1 \rangle \times \Pi_{i=j+1}^n \langle \alpha_i, \delta_i \rangle \right) \}. \end{aligned} \tag{3.5}$$

4 The main algorithm

Now we propose the solution algorithm for linear mixed integer programming problem (P) with scenario constraints.

Algorithm 4.1. (The algorithm for linear mixed integer programming problem (P))

Step 0. (Initialization) Obtain the continuous optimal solution $\tilde{x}$ by solving the continuous relaxation problem of (MP) associated with box $X = \langle l, u \rangle$. Set $f_{best} := +\infty,\ LB = f(\tilde{x})$, $X^0 := \{X\}$ and $k := 0$.

Step 1. Select the integer subbox $\langle \alpha^k, \beta^k \rangle$ from X^k that yields the minimum lower bound LB.

- If $\tilde{x}$ is an integer solution, then solve each $s \in \{1, 2, \cdots, S\}$ subproblems (SP_s) for given $\tilde{x}$. If all subproblems are feasible, then we obtain the optimal solution $\tilde{x}$ with the optimal objective value $f(\tilde{x})$ in the integer box $\langle \alpha^k, \beta^k \rangle$. And set $f_{best} := f(\tilde{x})$ and $x_{best} := \tilde{x}$ if $f(\tilde{x}) < f_{best}$. Specially, if $\langle \alpha^k, \beta^k \rangle = X$, stop and $x_{best} := \tilde{x}$ is the optimal solution of (P). Otherwise, solve problem (DP_s) for the infeasible subproblems (SP_s) and goto (i) of Step 2.
- If $\tilde{x}$ is not an integer solution, we can get two integer points $\tilde{x}^+$ and $\tilde{x}^-$ by the following equations, then goto (ii) of Step 2.

$$(\tilde{x}^+)_j = \begin{cases} \lfloor \tilde{x}_j \rfloor, & c_j < 0 \\ \lceil \tilde{x}_j \rceil, & c_j > 0 \end{cases}, \quad j = 1, 2, \cdots, n$$

$$(\tilde{x}^-)_j = \begin{cases} \lfloor \tilde{x}_j \rfloor, & c_j > 0 \\ \lceil \tilde{x}_j \rceil, & c_j < 0 \end{cases}, \quad j = 1, 2, \cdots, n$$

Step 2. (Domain partition)

(i) For given integer solution $\tilde{x}$ and some $s \in \{1, 2, \cdots, S\}$, if problem (SP_s) is infeasible, then calculate integer box B_2 via (3.2), cut it off from the box $\langle \alpha^k, \beta^k \rangle$.

(ii) Cut the domain according to the feasibility of $\tilde{x}^+$ and $\tilde{x}^-$:

- If $\tilde{x}^+$ is a feasible point, set $f_{best} := f(\tilde{x}^+)$ and $x_{best} := \tilde{x}^+$ if $f(\tilde{x}^+) < f_{best}$. Calculate integer box B_1 via (3.1) in the box $\langle \alpha^k, \beta^k \rangle$.
- If $\tilde{x}^+$ is an infeasible point with $D_i \tilde{x}^+ > d_i$, where $D_i = (d_{i1}, d_{i2}, \ldots, d_{in})$. Calculate integer box B_3 via (3.3) in the box $\langle \alpha^k, \beta^k \rangle$.
- If $\tilde{x}^+$ is an infeasible point to some $s \in \{1, 2, \cdots, S\}$ scenario constraint, calculate integer box B_2 via (3.2) similar to (i).
- For $\tilde{x}^-$, calculate integer box B_4 via (3.4).

Set $Y^{k+1} = [X^k \setminus B_4] \setminus B_1$ where B_1 can be B_2 or B_3 determined by the feasibility of $\tilde{x}^+$.

Step 3. For each new generated integer subbox $\langle \alpha, \beta \rangle \in Y^{k+1}$, solve its continuous relaxation problem of (MP), then obtain a continuous optimal solution $\tilde{x}$ and the corresponding lower bound $LB = f(\tilde{x})$. Remove from Y^{k+1} these integer boxes for which one of the following conditions holds:

- There are no feasible solutions.
- $LB \geq f_{best}$.

Set $X^{k+1} = Y^{k+1} \bigcup (X^k \setminus \langle \alpha^k, \beta^k \rangle)$

Step 4. (Termination) If X^{k+1} is empty, then stop and x_{best} is an optimal solution to (P). Otherwise, set $k := k + 1$, goto Step 1.

Theorem 4.2. *Algorithm 4.1 terminates at an optimal solution of (P) within a finite number of iterations.*

Proof. By domain partition in Step 2, no optimal solution is removed. Therefore, x_{best} must be the optimal solution to (P) when the algorithm stops in Step 4 with $X^{k+1} = \emptyset$. The finite termination of the algorithms is obvious due to the finiteness of X and the fact that at least $\tilde{x}^+$ and $\tilde{x}^-$ can be removed at each iteration. $\square$

5 Computational experiments

The algorithm is programmed in FORTRAN 90 and runs on a PC with Pentium(R) Dual-core CPU E6700@3.2GHz for problem (P).

10 problems with randomly generated data from uniform distribution are tested. $c \in [-10, 20]$, $T^s = (t_{ij})_{m_1 \times n}$ is generated with $t_{ij} \in [-1, 1]$ for $i = 1, \ldots, m_1,\ j = 1, \ldots, n$, $W^s = (w_{ij})_{m_1 \times n_1}$ with $w_{ij} \in [-5.5, 4.5]$ for $i = 1, \ldots, m_1,\ j = 1, \ldots, n_1$, and $h^s = T^s \cdot [l + rr \cdot (u - l)]$ for each scenario $s \in \{1, 2 \cdots, S\}$ where $S = 5$ and $rr \in [0.65, 0.95]$. The constraint matrix $D = (d_{ij})_{m \times n}$ is generated with $d_{ij} \in [-20, 20]$ for $i = 1, \ldots, m,\ j = 1, \ldots, n$; The right-hand side d is taken as $d = D \cdot [l + r \cdot (u - l)]$, where $l = (l_1, \ldots, l_n)^T$, $u = (u_1, \ldots, u_n)^T$, $l_i = 1,\ u_i = 5$ for $i = 1, \ldots, n$ and $r = 0.6$.

In our implementation, the continuous relaxation problem of (MP) is solved by the simplex method. The numerical results are summarized in Tables 1-6, where

- n = number of integer variables,
- n_1 = number of continuous variables in the scenario constraints,
- m = number of constraints $Dx \leq d$,
- S = number of scenario constraints,
- m_1 = number of constraints $T^s x + W^s y^s \geq h^s$,
- cut1 stands for number of feasible cut,
- cut2 stands for number of infeasible cut for scenario constraints,
- cut3 stands for number of infeasible cut for $Dx \leq d$,
- iters stands for number of iterations,
- brans stands for number of branches generated when solve the problems by the traditional branch-and-bound method,
- avg stands for average results of the algorithms for 10 test problems.

Table 1-3 summarize the numerical results for different size of scenario constraints, respectively. From Table 1-3, we can see that the average CPU time, the average integer boxes and the average iterations to solve the problems are increasing when the sizes of the problems are increasing for fixed size of scenario constraints. Also the problems with a large size of scenario constraints are more difficult to solve than the ones with a small size of scenario constraints in terms of the average CPU time. In Table 1, we can calculate the problems with $m = 30$, but we can only calculate the problems with $m = 25$ in Table 2 and $m = 18$ in Table 3. From the computational results in Table 1-3, we can observe that Algorithm 4.1 can find the exact solutions of linear mixed integer programming problems with scenario constraints in reasonable computation time.

On the other hand, we know that problem (P) can be solved by the traditional branch-and-bound method. The traditional branch-and-bound method is based on the continuous

Table 1: Numerical results for problem (P) with $n_1 = 10, m_1 = 5$

$n \times m$	Avg CPU	Avg boxes	Avg iters	Avg cut1	Avg cut2	Avg cut3
50×10	1.017	4061.0	164.9	19.9	40.5	104.1
110×10	31.175	43874.7	1083.2	23.0	705.6	354.6
160×10	92.041	58204.5	731.7	101.0	104.7	526.0
200×10	79.591	56381.7	1546.1	11.9	928.1	606.0
50×15	16.134	31570.1	1068.5	33.5	127.5	907.5
100×15	196.694	107788.7	1719.8	148.0	15.3	1556.4
150×15	496.361	151836.7	1527.9	240.3	59.1	1228.5
200×15	937.125	232691.6	2283.9	409.9	381.6	1492.4
50×20	224.659	228800.1	6812.5	79.2	23.6	6709.6
70×20	1412.874	975859.6	19404.4	243.2	24.6	19136.5
90×20	1417.124	859469.1	14546.7	210.5	41.8	14294.4
40×25	199.733	290690.3	14051.0	24.1	762.1	13264.8
50×25	1296.070	928937.2	26250.7	254.7	87.2	25908.8
70×25	1287.228	488210.1	10658.4	320.4	242.6	10095.3
35×30	352.392	318777.9	12199.3	45.8	40.9	12112.6
45×30	694.641	372077.4	10820.6	91.9	20.9	10707.8

optimal solution from the continuous relaxation problem of (P). In the traditional branch-and-bound method, according to an non-integer component x_i of the continuous optimal solution x, two branches, $x_i \leq \lfloor x_i \rfloor$ and $x_i \geq \lceil x_i \rceil$, are generated at each node, then add these two constraints to the original problem, respectively. Thus two subproblems are generated and one of them is selected to solve next. The performance of Algorithm 4.1 has been compared with the traditional branch-and-bound method and the comparison results are presented in Table 4-6 where average CPU time, average subbox number (or average branches) and average iterations are obtained by running 10 test problems for each $n \times m$. From Table 4-6, it is clear that the proposed algorithm is much better than the traditional branch-and-bound method in terms of average CPU time. This main reason is the innovation of the presented algorithm which lies in the decomposition method and the domain partition techniques, which diminish the dimension of problem (P) and the optimality gap gradually in the iteration process. According to our computational experiment, the CPU time used by Algorithm 4.1 depends both on the number of integer boxes and the computation time to identify the optimal solution to the continuous programming problem (MP) in each integer box. While the size of the problem in the tradition branch-and-bound method is very large since the number of the variables is $n + n_1 \times S$ and the number of constraints is $m + m_1 \times S$ except the lower bounds and the upper bounds, and the integer limitation, of the variables, thus it will spend much more CPU time to solve the continuous relaxation problem by the traditional branch-and-bound method. This is also witnessed by Table 4-6.

6 Conclusion

A new exact solution method is developed in this paper for solving linear mixed integer programming problems (P) with scenario constraints. The primal problem is divided into two subproblems by the decompose method, then some integer boxes that do not contain any optimal solution are cut off from X via domain partition techniques. Incorporating the solution scheme into a branch-and-bound frame, the proposed solution method reduces

Table 2: Numerical results for problem (P) with $n_1 = 13, m_1 = 7$

$n \times m$	Avg CPU	Avg boxes	Avg iters	Avg cut1	Avg cut2	Avg cut3
50×10	1.780	7425.7	454.3	18.3	248.9	187.0
100×10	11.155	16088.9	368.1	7.8	178.6	181.5
150×10	59.153	41637.7	563.2	26.1	245.5	291.6
200×10	72.892	33056.0	318.4	31.3	36.3	250.4
50×15	30.380	49877.8	1958.6	36.6	98.2	1823.8
100×15	267.408	230649.2	4916.9	47.7	343.6	4525.6
150×15	401.075	150206.5	1837.9	44.9	329.9	1463.0
200×15	1817.731	391428.2	3081.4	240.7	139.7	2700.9
50×20	623.125	605758.8	15377.2	119.0	94.0	15164.2
60×20	443.466	385772.6	12319.0	72.2	1440.7	10806.0
75×20	1210.913	690093.8	14161.4	822.2	18.7	13320.4
40×25	168.134	166052.1	6269.5	45.6	208.6	6015.3
45×25	277.128	190392.6	5832.7	109.1	111.8	5611.6
55×25	580.892	365227.4	10106.9	231.0	144.7	9731.1
60×25	2811.833	1485885.9	39630.9	239.6	176.4	39214.9

Table 3: Numerical results for problem (P) with $n_1 = 15, m_1 = 10$

$n \times m$	Avg CPU	Avg boxes	Avg iters	Avg cut1	Avg cut2	Avg cut3
70×10	6.388	13401.2	459.2	13.2	201.7	244.2
100×10	18.244	25162.9	531.8	60.7	123.7	347.4
150×10	40.686	35457.9	583.8	41.9	301.8	240.1
200×10	22.780	12194.4	219.5	12.4	136.3	70.6
80×14	58.598	58159.0	1519.3	74.4	347.3	1097.6
100×14	361.780	307521.1	6047.5	40.3	418.3	5588.9
150×14	351.650	164465.8	2176.0	103.0	283.6	1789.4
200×14	845.192	182235.1	1533.3	135.1	36.9	1361.3
80×18	75.081	50149.9	1154.1	72.4	101.6	980.1
100×18	510.251	247254.2	4380.5	283.0	36.8	4060.7
150×18	817.096	235639.0	2508.8	229.6	30.5	2248.7
180×18	1273.598	261445.2	2633.5	213.1	53.5	2633.9

Table 4: The comparison results with $n_1 = 10,\ m_1 = 5$

$n \times m$	Algorithm 4.1			Tradition BB		
	Avg CPU(s)	Avg iters	Avg boxes	Avg CPU(s)	Avg iters	Avg brans
20×10	0.102	924.0	78.3	17.484	2310.2	2311.2
30×10	0.675	4114.9	251.4	109.994	7609.8	7610.8
50×10	7.767	30731.2	1685.9	112.352	6261.2	6262.2
100×10	7.597	10046.3	283.3	608.692	17072.4	17073.4

Table 5: The comparison results with $n_1 = 13,\ m_1 = 7$

$n \times m$	Algorithm 4.1			Tradition BB		
	Avg CPU(s)	Avg iters	Avg boxes	Avg CPU(s)	Avg iters	Avg brans
20×10	0.652	6128.5	468.4	32.498	2529.6	2530.6
30×10	0.838	5447.0	304.9	373.977	19846.2	19847.2
50×10	1.917	7425.7	454.3	474.630	15897.4	15898.4
100×10	3.434	4134.2	71.0	1124.327	18933.4	18934.4

Table 6: The comparison results with $n_1 = 15,\ m_1 = 10$

$n \times m$	Algorithm 4.1			Tradition BB		
	Avg CPU(s)	Avg iters	Avg boxes	Avg CPU(s)	Avg iters	Avg brans
18×10	0.116	285.7	27.7	23.028	946.6	947.6
29×10	0.656	3987.4	222.3	177.316	7888.6	7889.6
51×10	2.353	8291.2	500.8	451.000	11889.0	11890.0
100×10	43.408	61653.2	1454.8	1354.208	13491.6	13492.6

the optimality gap gradually in the solution iterations. Furthermore, the proposed solution method is of a finite-step convergence. In contract to the traditional branch-and-bound method, the proposed method is more efficient. On one hand, the decompose method reduces the dimension of large-scale problem (P). On the other hand, the proposed method generates and evaluates more than two new integer boxes simultaneously, while two new subproblems are generated at each node and one of them is selected to solve next in the traditional branch-and-bound method. This exact algorithm can also be extended to solve nonlinear integer programming problems with scenario constraints.

References

[1] M.S. Bazaraa, H.D. Sherali and C.M. Shetty, *Nonlinear Programming: Theory and Algorithms*, John. Wiley and Sons, Inc., New York, 1993.

[2] A. Beck and M. Teboulle, Global optimality conditions for quadratic optimization problems with binary constraints, *SIAM J. Optimiz.* 11 (2000) 179–188.

[3] R. Bellman and S.E. Dreyfus, *Applied Dynamic Programming*, Princeton University Press, Princeton, 1962.

[4] H.P. Benson and S.S. Erengue, An algorithm for concave integer minimization over a polyhedron, *Nav. Res. Log.* 37 (1990) 515–525.

[5] H.P. Benson, S.S. Erengue and R. Horst, A note on adopting methods for continuous global optimization to the discrete case, *Ann. Oper. Res.* 25 (1990) 243–252.

[6] K.M. Bretthauer, A.V. Cabot and M.A. Venkataramanan, An algorithm and new penalties for concave integer minimization over a polyhedron, *Nav. Res. Log.* 41 (1994) 435–454.

[7] K.M. Bretthauer and B. Shetty, The nonlinear resource allocation problem, *Oper. Res.* 43 (1995) 670–683.

[8] K.M. Bretthauer and B. Shetty, The nonlinear knapsack problem–algorithms and applications, *Eur. J. Oper. Res.* 138 (2002) 459–472.

[9] A.V. Cabot and S.S. Erengue, A branch and bound algorithm for solving a class of nonlinear integer programming problems, *Nav. Res. Log.* 33 (1986) 559–567.

[10] M.W. Cooper, Survey of methods of pure nonlinear integer programming, *Manage. Sci.* 27 (1981) 353–361.

[11] G.B. Dantzig, Discrete variable extremum problems, *Oper. Res.* 5 (1957) 266–277.

[12] M. Held and R.M. Karp, A dynamic programming approach to sequencing problems, *J. Soc. Ind. Appl. Math.* 10 (1962) 196–210.

[13] D. Hochbaum, A nonlinear knapsack problem, *Oper. Res. Lett.* 17 (1995) 103–110.

[14] R. Horst, P.M. Pardalos and N.V. Thoai, *Introduction to Global Optimization, Second Edition, Nonconvex optimization and its application*, Kluwer, Dordrecht, 2000.

[15] R. Horst and N.V. Thoai, An integer concave minimization approach for the minimum concave cost capacitated flow problem on networks, *OR Spektrum* 20 (1998) 47–53.

[16] R.A. Howard, Dynamic programming, *Manage. Sci.* 12 (1966) 317–348.

[17] T. Ibaraki and N. Katoh, *Resource Allocation Problems: Algorithmic Approaches*, MIT Press, Cambridge, 1988.

[18] M.S. Kodialam and H. Luss, Algorithm for separable nonlinear resource allocation problems, *Oper. Res.* 46 (1998) 272–284.

[19] D. Li and X.L. Sun, *Nonlinear Integer Programming*, Springer Press, New York, 2006.

[20] D. Li, X.L. Sun and F.L. Wang, Convergent Lagrangian and contour cut method for nonlinear integer programming with a quadratic objective function, *SIAM J. Optim.* 17 (2006) 372–400.

[21] D. Li, X.L. Sun, J. Wang and K.I.M. McKinnon, Convergent Lagrangian and domain cut method for nonlinear knapsack problem, *Comput. Optim. Appl.* 42 (2009) 67–104.

[22] R.E. Marsten and T.L. Morin, A hybrid approach to discrete mathematical programming, *Math. Program.* 14 (1978) 21–40.

[23] K. Mathur, H.M. Salkin and S. Morito, A branch and search algorithm for a class of nonlinear knapsack problems, *Oper. Res. Lett.* 2 (1983) 55–60.

[24] T.L. Morin and R.E. Marsten, An algorithm for nonlinear knapsack problems, *Manage. Sci.* 22 (1976) 1147–1158.

[25] T.L. Morin and R.E. Marsten, Branch and bound strategies for dynamic programming, *Oper. Res.* 24 (1976) 611–627.

[26] P.M. Pardalos and N. Kovoor, An algorithm for a singly constrained class of quadratic programs subject to upper and lower bounds, *Math. Program.* 46 (1990) 321–328.

[27] P.M. Pardalos and J.B. Rosen, Reduction of nonlinear integer separable programming problems, *Int. J. Comput. Math.* 24 (1988) 55–64.

[28] X.L. Sun and D. Li, Optimality condition and branch and bound algorithm for constrained redundancy optimization in series systems, *Optim. Eng.* 3 (2002) 53–65.

[29] F.L. Wang, A new exact algorithm for concave knapsack problems with integer variables, *Int. J. Comput. Math.* 96 (2019) 126-134.

[30] F.L. Wang, A successive domain-reduction scheme for linearly constrained quadratic integer programming problems, *J. Oper. Res. Soc.* 72 (2020) 2317-2330.

[31] F.L. Wang, A combination of linear approxination and Lagrangian dual with a simple cut for general separable integer programming problems, *Pac. J. Optim.* 16 (2020) 441-459.

[32] F.L. Wang and L.Y. Cao, A new algorithm for quadratic integer programming problems with cardinality constraint, *Jpn. J. Ind. Appl. Math.* 37 (2020) 449-460.

Manuscript received 10 February 2023
revised 21 September 2023
accepted for publication 10 October 2023

Fenlan Wang
College of Science, Nanjing University of Aeronautics and Astronautics
29 Yudao St., Nanjing 210016, P.R. China
E-mail address: flwang@nuaa.edu.cn